日　本　国　特　許　庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日
Date of Application: ２００３年　３月１７日

出願番号
Application Number: 特願２００３－０７１８６７

[ST.10/C]: [ＪＰ２００３－０７１８６７]

出願人
Applicant(s): ジーイー・メディカル・システムズ・グローバル・テクノロジー・カンパニー・エルエルシー

２００３年　４月２２日

特許庁長官
Commissioner,
Japan Patent Office 太田信一郎

出証番号　出証特２００３－３０２９７２６

【書類名】　特許願
【整理番号】　16IN-I0224
【提出日】　平成15年 3月17日
【あて先】　特許庁長官殿
【国際特許分類】　G06F 15/00
【発明の名称】　マルチティア・アプリケーション・アーキテクチャ
【請求項の数】　7
【発明者】
【住所又は居所】　インド国、バンガロール、ホワイトフィールド　ロード、ジェーエフダブリューティーシー、８００　ビジネスセンター　ドライブ
【氏名】　アービシュカール　バーララ
【特許出願人】
【識別番号】　300019238
【氏名又は名称】　ジーイー・メディカル・システムズ・グローバル・テクノロジー・カンパニー・エルエルシー
【代理人】
【識別番号】　100085187
【弁理士】
【氏名又は名称】　井島　藤治
【選任した代理人】
【識別番号】　100090424
【弁理士】
【氏名又は名称】　鮫島　信重
【手数料の表示】
【予納台帳番号】　009542
【納付金額】　21,000円
【提出物件の目録】
【物件名】　明細書　1

【物件名】　　図面　1

【物件名】　　要約書　1

【包括委任状番号】　0005611

【プルーフの要否】　　要

【書類名】　明細書

【発明の名称】　マルチティア・アプリケーション・アーキテクチャ

【特許請求の範囲】

【請求項１】　ミドルティアを有するマルチティア・アプリケーション・アーキテクチャであって、

アプリケーションとミドルティアを仲介するフレームワークを具備し、

前記フレームワークは、

アプリケーションがキャッシュから取り出したオブジェクトをミドルティアに実行させ、

オブジェクトがステイルしたときそのオブジェクトを予め定められた試行回数の限度内で繰り返しリフレッシュし、

リフレッシュが成功したときはそのオブジェクトをキャッシュに返すとともに再度ミドルティアに実行させ、

オブジェクトのリフレッシュが試行回数の限度内で成功しないときはアプリケーションをフェイルセーフ状態でクイットする、

ことを特徴とするマルチティア・アプリケーション・アーキテクチャ。

【請求項２】　前記試行回数の限度はユーザーにより設定可能である、

ことを特徴とする請求項１に記載のマルチティア・アプリケーション・アーキテクチャ。

【請求項３】　前記試行の時間間隔はユーザーにより設定可能である、

ことを特徴とする請求項２に記載のマルチティア・アプリケーション・アーキテクチャ。

【請求項４】　前記フレームワークは、その動作がユーザーに対して可視化されている、

ことを特徴とする請求項１ないし請求項３のうちのいずれか１つに記載のマルチティア・アプリケーション・アーキテクチャ。

【請求項５】　前記ミドルティアがクラッシュしたときに正常動作を回復させるウォッチドッグを具備する、

ことを特徴とする請求項１ないし請求項４のうちのいずれか１つに記載のマルチ

ティア・アプリケーション・アーキテクチャ。

【請求項６】　前記ウォッチドッグは定期的なポーリングの結果に基づいてミドルティアを回復させる、

ことを特徴とする請求項５に記載のマルチティア・アプリケーション・アーキテクチャ。

【請求項７】　前記ウォッチドッグは前記フレームワークからの通知に基づいてミドルティアを回復させる、

ことを特徴とする請求項５に記載のマルチティア・アプリケーション・アーキテクチャ。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、マルチティア・アプリケーション・アーキテクチャ（ｍｕｌｔｉ－ｔｉｅｒ　ａｐｐｌｉｃａｔｉｏｎ　ａｒｃｈｉｔｅｃｔｕｒｅ）に関し、とくに、アプリケーション・サーバ（ａｐｐｌｉｃａｔｉｏｎ　ｓｅｒｖｅｒ）やウェブ・サーバ（Ｗｅｂ　ｓｅｒｖｅｒ）のようなミドルティア（ｍｉｄｄｌｅ　ｔｉｅｒ）を有するマルチティア・アプリケーション・アーキテクチャに関する。

【０００２】

【従来の技術】

マルチティア・アプリケーション・アーキテクチャでは、アプリケーション用のビジネスロジック（ｂｕｓｉｎｅｓｓ　ｌｏｇｉｃ）は、ミドルティアにおいて実行される。ビジネスロジックはアプリケーションからミドルティアにオブジェクト（ｏｂｊｅｃｔ）として指定され、ミドルティアは指定されたオブジェクトを実行する（例えば、非特許文献１，２参照）。ミドルティア内のオブジェクトへのアクセス（ａｃｃｅｓｓ）はサービスロケータ（ｓｅｒｖｉｃｅ　ｌｏｃａｔｏｒ）を通じて行われる（例えば、非特許文献３参照）。

【０００３】

【非特許文献１】

スティーブン グールド（Ｓｔｅｖｅｎ Ｇｏｕｌｄ）、”ディベロップ エヌティア アプリケーションズ ユージング ジェイ２イーイー（Ｄｅｖｅｌｏｐ ｎ－ｔｉｅｒ ａｐｐｌｉｃａｔｉｏｎｓ ｕｓｉｎｇ Ｊ２ＥＥ）”、図２、「ｏｎｌｉｎｅ」、２００２年、ジャバワールド・ドットコム（ＪａｖａＷｏｒｌｄ．ｃｏｍ）、ジャバワールド（Ｊａｖａ Ｗｏｒｌｄ）、ｐ．３／１０、「平成１５年１月２３日検索」、インターネット＜ＵＲＬ：ｈｔｔｐ：／／ｗｗｗ．ｊａｖａｗｏｒｌｄ．ｃｏｍ／ｊａｖａｗｏｒｌｄ／ｊｗ－１２－２０００／ｊｗ－１２０１／ｗｅｂｌｏｇｉｃ＿ｐ．ｈｔｍｌ＞

【非特許文献２】

リチャード ジー ボールドウィン（Ｒｉｃｈａｒｄ Ｇ．Ｂａｌｄｗｉｎ）、”エンタープライズ ジャバビーンズ： ミドルティア サーバーズ アンド ジェイ２イーイー（Ｅｎｔｅｒｐｒｉｓｅ ＪａｖａＢｅａｎｓ：Ｍｉｄｄｌｅ－Ｔｉｅｒ Ｓｅｒｖｅｒｓ ａｎｄ Ｊ２ＥＥ）”、見出し：アミドルティア サーバ（Ａ Ｍｉｄｄｌｅ－Ｔｉｅｒ Ｓｅｒｅｒ）、「ｏｎｌｉｎｅ」、２００２年、ジュピターメデイア コーポレーション（Ｊｕｐｉｔｅｒｍｅｄｉａ Ｃｏｒｐｏｒａｔｉｏｎ）、ディベロッパー ドットコム（ｄｅｖｅｌｏｐｅｒ．ｃｏｍ ＧＡＭＥＬＡＮ）、ｐ．４／１２－５／１２、「平成１４年１２月１１日検索」、インターネット＜ＵＲＬ：ｈｔｔｐ：／／ｗｗｗ．ｄｅｖｅｌｏｐｅｒ．ｃｏｍ／ｊａｖａ／ｏｔｈｅｒ／ａｒｔｉｃｌｅ．ｐｈｐ／６４１３３１＞

【非特許文献３】

”サン ジャバ センター ジェイ２イーイー パターンズ サービス ロケータ（Ｓｕｎ Ｊａｖａ Ｃｅｎｔｅｒ Ｊ２ＥＥ Ｐａｔｔｅｒｎｓ Ｓｅｒｖｉｃｅ Ｌｏｃａｔｏｒ）、見出し：ソリューション（Ｓｏｌｕｔｉｏｎ）、「ｏｎｌｉｎｅ」、２００２年、サン マイクロシステムズ インコーポレーテッド（Ｓｕｎ Ｍｉｃｒｏｓｙｓｔｅｍｓ，Ｉｎｃ．）、ジャバテクノロジー ホームページ（Ｊａｖａ Ｔｅｃｈｏｎｏｌｏｇｙ Ｈｏｍｅ Ｐａｇｅ）、ｐ．３／１２－６／１２、「平成１４年１２月２４日検索」、インターネット＜ＵＲＬ：ｆｉｌｅ：／／Ｃ：￥ＷＩＮＤＯＷＳ￥ＴＥＭＰ￥ＴＤ＿００２

４．ＤＩＲ￥Ｓｕｎ％２０Ｊａｖａ％２０Ｃｅｎｔｅｒ％２０－％３０Ｓｅｒｖｉｃｅ％２０Ｌｏｃａｔｏｒ％２０Ｊ２ＥＥ％３０Ｐａｔｔｅｒｎｓ．ｈｔｍ

【０００４】

【発明が解決しようとする課題】

上記のようなマルチティア・アプリケーション・アーキテクチャでは、ミドルティア内のオブジェクトへのアクセスがサービスロケータを通じて行われるのでアプリケーションの動作が遅くなる。

【０００５】

また、ミドルティアがクラッシュすると、オブジェクトがステイル（ｓｔａｌｅ）することによりアプリケーションがハングアップ（ｈａｎｇ－ｕｐ）するが、そのような場合についてのフェイルセーフ（ｆａｉｌ　ｓａｆｅ）対策がなされていないので、アプリケーションのハングアップによって、他のアプリケーションへの妨害等、二次的な障害が発生するおそれがある。

【０００６】

そこで、本発明の課題は、オブジェクトへのアクセスが高速でかつミドルティアのクラッシュに対するフェイルセーフ機能を備えたマルチティア・アプリケーション・アーキテクチャを実現することである。なお、本書において、アーキテクチャとはコンピュータ・プログラム（ｃｏｍｐｕｔｅｒ　ｐｒｏｇｒａｍ）のアーキテクチャを意味する。

【０００７】

【課題を解決するための手段】

上記の課題を解決するための本発明は、ミドルティアを有するマルチティア・アプリケーション・アーキテクチャであって、アプリケーションとミドルティアを仲介するフレームワークを具備し、前記フレームワークは、アプリケーションがキャッシュから取り出したオブジェクトをミドルティアに実行させ、オブジェクトがステイルしたときそのオブジェクトを予め定められた試行限度内で繰り返しリフレッシュし、リフレッシュが成功したときはそのオブジェクトをキャッシュに返すとともに再度ミドルティアに実行させ、オブジェクトのリフレッシュが試行限度内で成功しないときはアプリケーションをフェイルセーフ状態でクイッ

トする、ことを特徴とするマルチティア・アプリケーション・アーキテクチャである。

【０００８】

本発明では、アプリケーションとミドルティアを仲介するフレームワークにより、アプリケーションがキャッシュから取り出したオブジェクトをミドルティアに実行させ、オブジェクトがステイルしたときそのオブジェクトを予め定められた試行回数の限度内で繰り返しリフレッシュし、リフレッシュが成功したときはそのオブジェクトをキャッシュに返すとともに再度ミドルティアに実行させ、オブジェクトのリフレッシュが試行回数の限度内で成功しないときはアプリケーションをフェイルセーフ状態でクイットするようにしたので、オブジェクトへのアクセスが高速でかつミドルティアのクラッシュに対するフェイルセーフ機能を備えたマルチティア・アプリケーション・アーキテクチャを実現することができる。

【０００９】

前記試行回数の限度はユーザーにより設定可能であることが、ニーズに合致した試行限度とする点で好ましい。前記試行の時間間隔はユーザーにより設定可能であることが、ニーズに合致した試行間隔とする点で好ましい。前記フレームワークは、その動作がユーザーに対して可視化されていることが、ユーザーによる状況把握を容易にする点で好ましい。

【００１０】

前記ミドルティアがクラッシュしたときに正常動作を回復させるウォッチドッグを具備することが、回復を自動化する点で好ましい。前記ウォッチドッグは定期的なポーリングの結果に基づいてミドルティアを回復させることが、ウォッチドッグ主導のクラッシュ回復を行う点で好ましい。前記ウォッチドッグは前記フレームワークからの通知に基づいてミドルティアを回復させることが、フレームワーク主導のクラッシュ回復を行う点で好ましい。

【００１１】

【発明の実施の形態】

以下、図面を参照して本発明の実施の形態を詳細に説明する。なお、本発明は

実施の形態に限定されるものではない。図１に、マルチティア・アプリケーション・アーキテクチャの構成をブロック（ｂｌｏｃｋ）図によって示す。本アーキテクチャは本発明の実施の形態の一例である。本アーキテクチャの構成によって、本発明のマルチティア・アプリケーション・アーキテクチャに関する実施の形態の一例が示される。

【００１２】

図１に示すように、本アーキテクチャは、フロントエンドティア（ｆｒｏｎｔ－ｅｎｄ　ｔｉｅｒ）２、ミドルティア４およびバックエンドティア（ｂａｃｋ－ｅｎｄ　ｔｉｅｒ）６を有する。

【００１３】

フロントエンドティア２はクライアント（ｃｌｉｅｎｔ）に相当する。ミドルティア４はアプリケーションサーバに相当する。バックエンドティア６は例えばデータベースサーバ（ｄａｔａｂａｓｅ　ｓｅｒｖｅｒ）等のＥＩＳ（ｅｎｔｅｒｐｒｉｓｅ　ｉｎｆｏｒｍａｔｉｏｎ　ｓｅｒｖｉｃｅ）に相当する。なお、フロントエンドティア２とミドルティア４の間に、他のミドルティア例えばウェブサーバ（Ｗｅｂ　ｓｅｒｖｅｒ）等があってもよい。

【００１４】

フロントエンドティア２はバックエンドティア６の資源（例えば、データベース等）を利用するアプリケーションを有し、そのアプリケーションが必要とするビジネスロジックの実行がミドルティア４によってサービスされる。

【００１５】

フロントエンドティア２からミドルティア４へのビジネスロジックの実行要求はオブジェクトとして与えられる。オブジェクトは、フロントエンドティア２内のフレームワーク（ｆｒａｍｅｗｏｒｋ）２２を通じてミドルティア４に与えられる。

【００１６】

図２に、フレームワーク２２を含むフロントエンドティア２の主要部の構成をミドルティア４およびバックエンドティア６とともに示す。同図に示すように、フロントエンドティア２は、アプリケーション２０２およびキャッシュ（ｃａｃ

ｈｅ）２０４を有する。

【００１７】

キャッシュ２０４は、アプリケーション２０２が利用するオブジェクトを保持している。保持するオブジェクトは、アプリケーション２０２が利用する全てのオブジェクトまたは主要なオブジェクトである。オブジェクトはリモートレファレンス（ｒｅｍｏｔｅ　ｒｅｆｅｒｅｎｃｅ）として保持される。

【００１８】

アプリケーション２０２が利用する全てのオブジェクトまたは主要なオブジェクトのリモートレファレンスをキャッシュ２０４に保持するので、ホームレファレンス（ｈｏｍｅ　ｒｅｆｅｒｅｎｃｅ）を用いたリモートレファレンスの（ｌｏｏｋ－ｕｐ）ルックアップが不要になり、その分アプリケーションの動作は高速になる。

【００１９】

フレームワーク２２は、ロジックハンドラ（ｌｏｇｉｃ　ｈａｎｄｌｅｒ）２２２、ディテクタ（ｄｅｔｅｃｔｏｒ）２２４、リフレッシャ（ｒｅｆｒｅｓｈｅｒ）２２６およびクイッタ（ｑｕｉｔｔｅｒ）２２８を有する。

【００２０】

アプリケーション２０２は、キャッシュ２０４から１つのオブジェクトを取り出してロジックハンドラ２２２に供給する。ロジックハンドラ２２２はそのオブジェクトを用いて、ミドルティア４に、対応するビジネスロジックのインボーク（ｉｎｖｏｋｅ）を行う。

【００２１】

ミドルティア４はビジネスロジックを実行する。ビジネスロジックの実効状態および実行結果がディテクタ２２４によって検出され、ビジネスロジックのサクセス（ｓｕｃｃｅｓｓ）またはフェイル（ｆａｉｌ）がロジックハンドラ２２２に通知される。なお、サクセスはアプリケーション２０２にも通知される。

【００２２】

ビジネスロジックの実行がサクセスした場合は、ロジックハンドラ２２２は実行済みのオブジェクトをキャッシュに返却し、アプリケーション２０２は新たな

オブジェクトをキャッシュ２０４から取り出してロジックハンドラ２２２に供給する。ビジネスロジックの実行がサクセスする間は、以上のプロセス（ｐｒｏｃｅｓｓ）が繰り返される。

【００２３】

ビジネスロジックの実行がフェイルした場合は、オブジェクトがステイルする。このとき、ロジックハンドラ２２２はリフレッシャ２２６にステイルしたオブジェクトのリフレッシュを行わせる。リフレッシュは次のようにして行われる。

【００２４】

すなわち、リフレッシャ２２６は、オブジェクトのホームレファレンスを用いて、ミドルティア４に対してリモートレファレンスのルックアップを行う。

【００２５】

ルックアップが成功したときは、ミドルティア４からリフレッシャ２２６にリモートレファレンスが返される。これによって、ステイルしたオブジェクトのリフレッシュが行われる。そこで、ロジックハンドラ２２２はリフレッシュされたオブジェクトをキャッシュ２０４に返すとともに、そのオブジェクトについてあらためてインボークを行う。これによって、ステイルしたオブジェクトを復活させることが可能になる。

【００２６】

ルックアップが不成功のときはリモートレファレンスが帰ってこないので、ルックアップのやり直しが行われる。ルックアップの不成功が続く間、ルックアップのやり直しが行われる。すなわち、リフレッシュの試行が複数回繰り返される。これによって、オブジェクトのリフレッシュの成功率が向上する。

【００２７】

繰り返しの回数の上限および繰り返しの時間間隔は予め定められている。回数および間隔はユーザー（ｕｓｅｒ）が設定可能にすることが、ニーズに合致した回数および間隔とする点で好ましい。

【００２８】

繰り返し回数が上限に達してもリフレッシュが成功しないときは、ロジックハンドラ２２２はクイッタ２２６にフェイルセーフを行わせる。クイッタ２２６は

、アプリケーション２０２が専有していた各種の資源を解放すること等により、アプリケーション２０２についてフェイルセーフクイット（ｆａｉｌ　ｓａｆｅ　ｑｕｉｔ）を行う。このようなフェイルセーフ処理により、その後のアプリケーション２０２の再起動を円滑にすることができ、また、資源を共用する他のアプリケーションへの妨害をなくすることができる。

【００２９】

このようなフレームワーク２２は、エンティティビーンズ（ｅｎｔｉｔｙ　ｂｅａｎｓ）やステイトレス（ｓｔａｔｅｌｅｓｓ）のセッションビーンズ（ｓｅｓｓｉｏｎ　ｂｅａｎｓ）のハンドリングにとくに好適である。

【００３０】

以上のようなフレームワークの動作状況は、ユーザーが認識可能なものとされる。これは、適宜のＧＵＩ（ｇｒａｐｈｉｃａｌ　ｕｓｅｒ　ｉｎｔｅｒｆａｃｅ）等を利用して行われる。これによって、ユーザーによる状況把握を容易にすることができる。なお、必要がなければこれは省略してもよい。

【００３１】

ミドルティア４におけるビジネスロジックのフェイルはミドルティア４のクラッシュ（ｃｒｕｓｈ）等によって発生する。クラッシュが発生した場合でもそれを自動的に回復させることができれば、ミドルティア４の稼働率を良くすることができ、アプリケーションを効果的に実行することが可能になる。

【００３２】

そこで、図３に示すように、ウォッチドッグ（ｗａｔｃｈｄｏｇ）４０２を用いてミドルティア４のクラッシュの自動回復を行う。ウォッチドッグ４０２はポーリング（ｐｏｌｌｉｎｇ）を定期的に行ってミドルティア４のクラッシュの有無を監視し、クラッシュが発生したときはその回復作業を行う。このようなウォッチドッグ４０２は、例えばシェルスクリプト（ｓｈｅｌｌ　ｓｃｒｉｐｔ）等を用いて実現することができる。

【００３３】

ウォッチドッグ４０２によるミドルティア４の回復は、図４に示すように、ロジックハンドラ２２２からのクラッシュ通知に基づいて行うようにしてもよい。

このようなウォッチドッグ４０２は、専用のアプリケーション等によって実現することができる。なお、ミドルティア４のクラッシュは、ロジックハンドラ２２２がリフレッシュによるオブジェクトの復活ができなかったときに通知される。

【００３４】

【発明の効果】

以上詳細に説明したように、本発明によれば、オブジェクトへのアクセスが高速でかつミドルティアのクラッシュに対するフェイルセーフ機能を備えたマルチティア・アプリケーション・アーキテクチャを実現することができる。

【図面の簡単な説明】

【図１】

本発明の実施の形態の一例の構成を示すブロック図である。

【図２】

本発明の実施の形態の一例についてフロントエンドティアの構成をより詳細に示すブロック図である。

【図３】

本発明の実施の形態の一例についてフロントエンドティアの構成をより詳細に示すブロック図である。

【図４】

本発明の実施の形態の一例についてフロントエンドティアの構成をより詳細に示すブロック図である。

【符号の説明】

２　フロントエンドティア

２２　フレームワーク

４　ミドルティア

６　バックエンドティア

２０２　アプリケーション

２０４　キャッシュ

２２２　ロジックハンドラ

２２４　ディテクタ

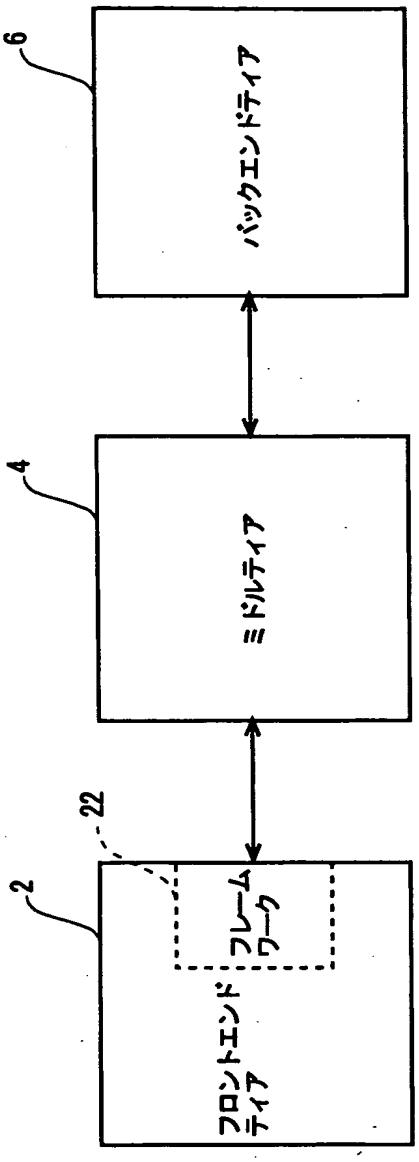
２２６　リフレッシャ

２２８　クイッタ

４０２　ウォッチドッグ

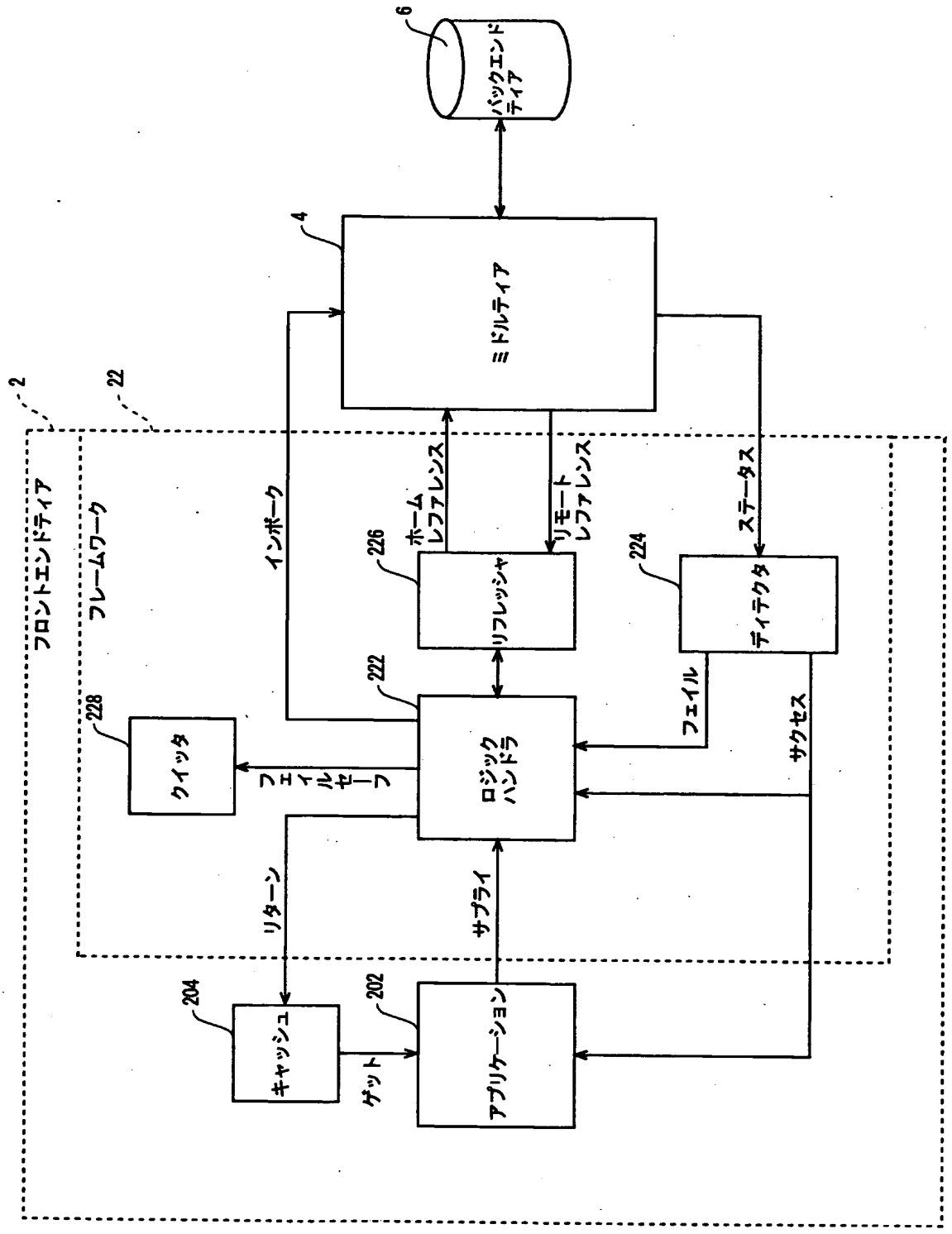
【書類名】　　図面

【図１】

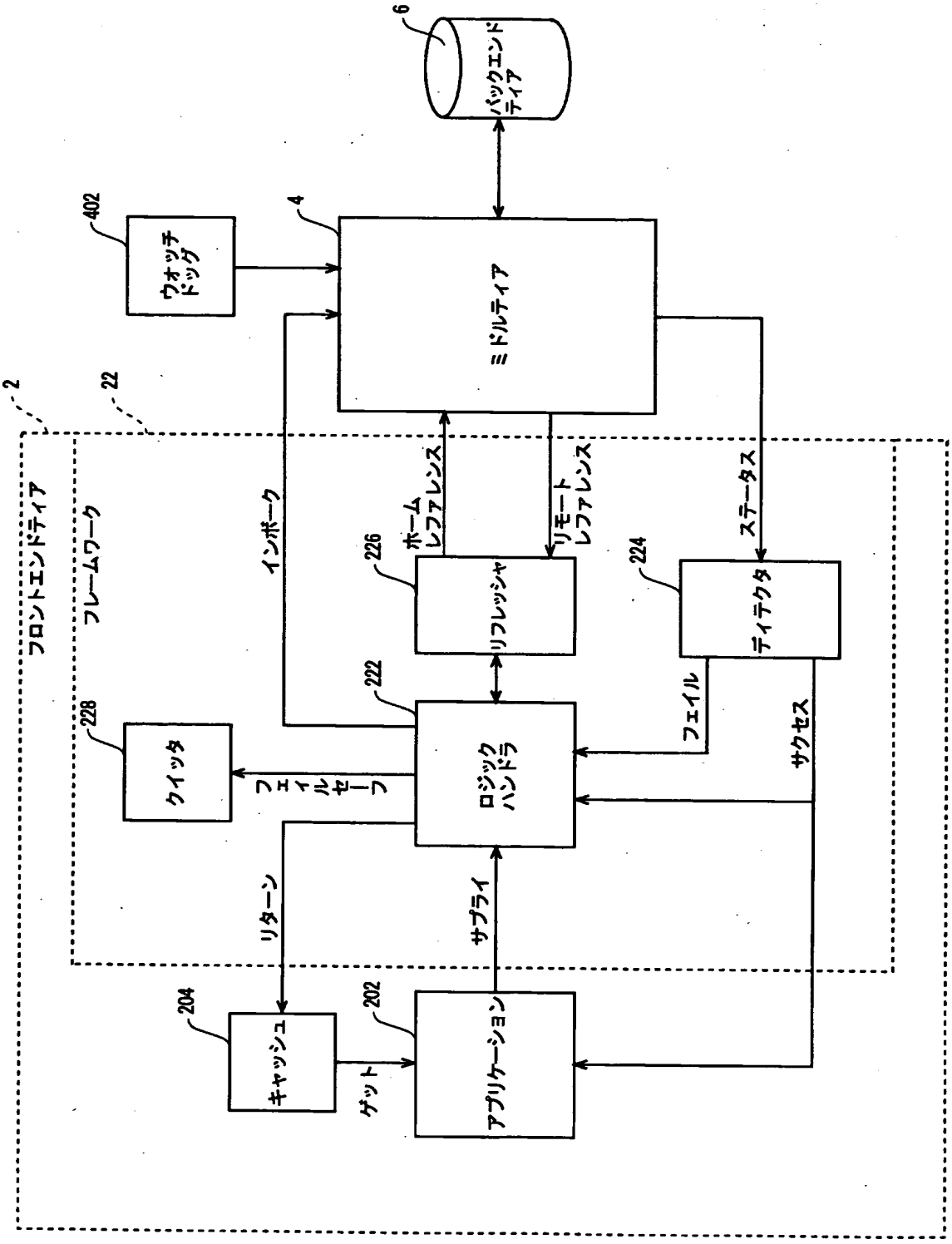


出証特２００３－３０２９７２６

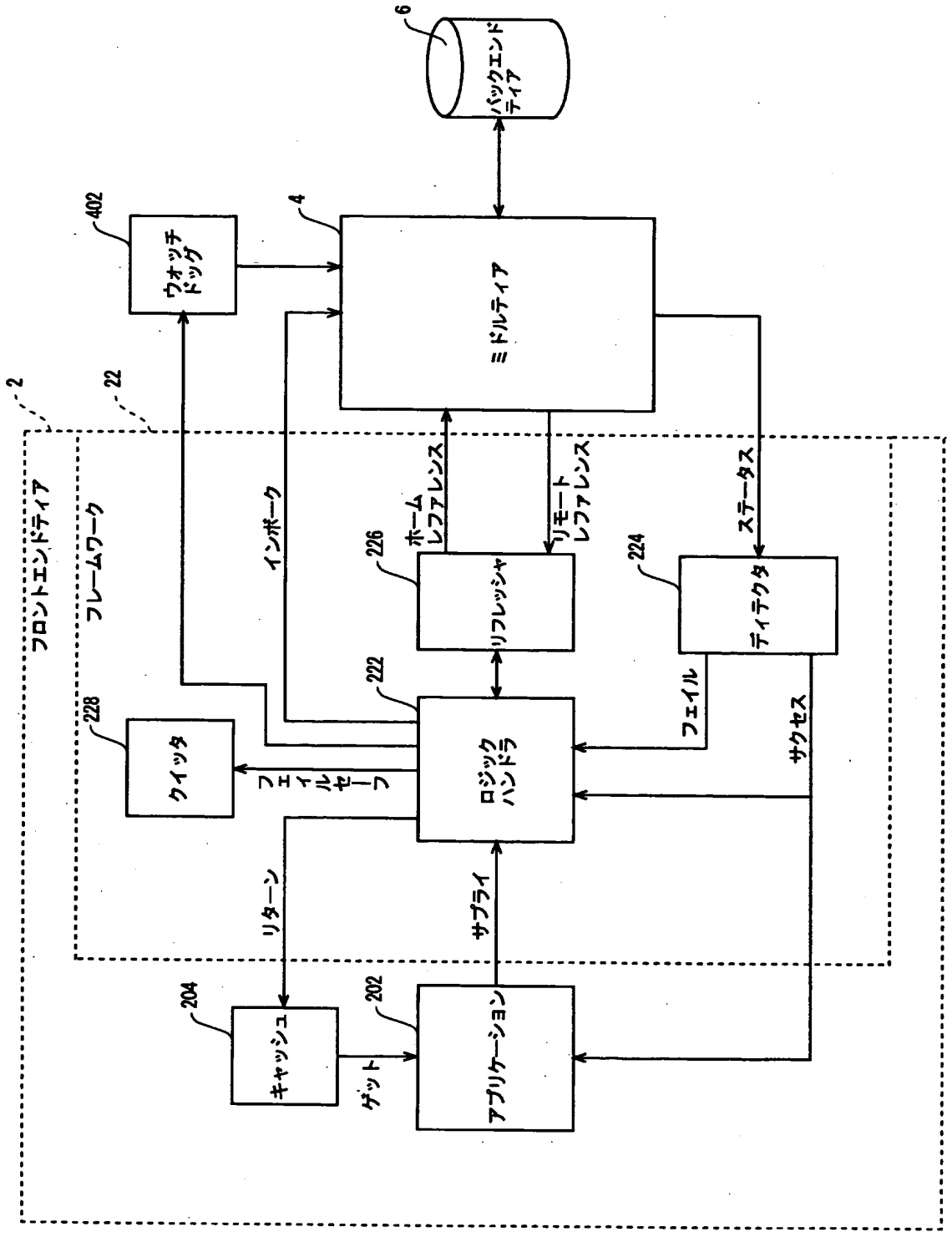
【図２】


6
バックエンドティア
4
ミドルティア
2
22
フロントエンドティア
フレームワーク
インボーク
ホームレファレンス
リモートレファレンス
ステータス
226
リフレッシャ
224
ディテクタ
222
ロジックハンドラ
228
クイッタ
フェイルセーフ
フェイル
サクセス
リターン
サプライ
204
キャッシュ
202
アプリケーション
ゲット

【図３】


6
バックエンドティア
402
ウォッチドッグ
4
ミドルティア
2
22
フロントエンドティア
フレームワーク
インボーク
ホームレファレンス
リモートレファレンス
ステータス
226
リフレッシャ
224
ディテクタ
222
ロジックハンドラ
228
クイッタ
フェイルセーフ
フェイル
サクセス
リターン
サプライ
204
キャッシュ
202
アプリケーション
ゲット

【図４】


6
バックエンドティア
402
ウォッチドッグ
4
ミドルティア
2
22
フロントエンドティア
フレームワーク
インボーク
ホームレファレンス
リモートレファレンス
ステータス
226
リフレッシャ
224
ディテクタ
222
ロジックハンドラ
228
クイッタ
フェイルセーフ
フェイル
サクセス
リターン
サプライ
204
キャッシュ
202
アプリケーション
ゲット

【書類名】　　要約書

【要約】

【課題】　オブジェクトへのアクセスが高速でかつミドルティアのクラッシュに対するフェイルセーフ機能を備えたマルチティア・アプリケーション・アーキテクチャを実現する。

【解決手段】　アプリケーション（２０２）とミドルティア（４）を仲介するフレームワーク（２２）により、アプリケーションがキャッシュ（２０４）から取り出したオブジェクトをミドルティアに実行させ（２２２）、オブジェクトがステイルしたときそのオブジェクトを予め定められた試行回数の限度内で繰り返しリフレッシュを試行し（２２６）、リフレッシュが成功したときはそのオブジェクトをキャッシュに返すとともに再度ミドルティアに実行させ、オブジェクトのリフレッシュが試行回数の限度内で成功しないときはアプリケーションをフェイルセーフ状態でクイットする（２２８）。

【選択図】　　図２

出　願　人　履　歴　情　報

識別番号　　〔３００００１９２３８〕

１．変更年月日　　２０００年　３月１５日
［変更理由］　名称変更
住　所　　アメリカ合衆国・ウィスコンシン州・５３１８８・ワウケシャ・ノース・グランドヴュー・ブールバード・ダブリュー・７１０・３０００
氏　名　　ジーイー・メディカル・システムズ・グローバル・テクノロジー・カンパニー・エルエルシー